Exercises for Chapter 11 of Vinod’s
“HANDS-ON INTERMEDIATE ECONOMETRICS USING R”

H. D. Vinod Professor of Economics,
Fordham University, Bronx, New York 10458

Abstract

These are exercises to accompany the above-mentioned book with the URL: http://www.worldscibooks.com/economics/6895.html
At this time all of the following exercises are suggested by H. D. Vinod (HDV) himself. Vinod invites readers to suggest new and challenging exercises (along with full and detailed answers) dealing with the discussion in Chapter 11 and related discussion from econometric literature by sending e-mail to vinod@fordham.edu. If the answers involve R programs, they must work. Readers may also suggest improvements to the answers and/or hints to existing exercises. If we include exercises and improvements suggested by readers, we promise to give credit to such readers by name. Furthermore, we will attach the initials of readers to individual exercises to identify the reader. Some R outputs are suppressed for brevity.

11 Exercises Mostly Based on Chapter 11 (Box-Cox, Loess, PPR) of the text

11.1 Exercise (Hedonic Price, Box-Cox)

Apply the Box-Cox transformation to the hedonic price model where the effect of crime on house prices is negative. Use the data called ‘Hedonic’ from the package ‘Ecdat’ [1]
Figure 1: Choice of the power $\lambda$ in Box-Cox transformation using log likelihood plot.

```r
library(Ecdat); data(Hedonic); attach(Hedonic)
reg=lm( mv~crim+nox+rm)
library(MASS)
boxcox(reg1, lambda = seq(-2, 1, len=20))
```

The plot of Box-cox likelihood is given in the attached figure. It suggests the choice of $\lambda = -0.78$.

### 11.2 Exercise (Scattplot smoothing Hedonic Price)

Apply scatterplot smoothing technique to the hedonic price model where the effect of crime on house prices is negative. Use the data called ‘Hedonic’ from the package ‘Ecdat.’
11.3 Exercise (Superiority of Loess Fit Rgressions for Forecasting)

Show that the loess method provides a superior fit to the data using the Icecream example used above.

```
library(Ecdat)
data(Icecream); attach(Icecream)
reg1=lm(cons~income+price+temp)
loe=loess(cons~income+price+temp)
pry=predict(loe)
ssloess=sum((cons-pry)^2)
ssreg=sum((resid(reg1)^2))
ssloess/ssreg #0.18
matplot(cbind(cons, pry, fitted(reg1)), typ="l",
main="Icecream consumption, OLS and LOESS fitted values")
matplot(cbind(cons, pry, fitted(reg1)), typ="l",
main="Icecream consumption, OLS and LOESS fitted values")
legend(x=3,y=0.5, legend=c("cons", "Loess Fit", "OLS Fit"),
1ty=1:3, col=1:3)
```

In the Icecream example the residual sum of squares by loess is only about 18% the residual sum of squares by OLS showing a superior fit. This is seen graphically below.

11.4 Exercise (Projection pursuit regression)

Describe how computer intensive projection pursuit regression (ppr) handles the curse of dimensionality and how Vinod’s Theorem 2 is helpful to estimate partial derivatives $\partial E(y|X)/\partial X$ despite nonlinearity of the underlying relations and without adding to the number of parameters. (Hint: See Sec. 11.4 of the text [3].)

Further hints for an ANSWER: The PPR produces estimates of $\gamma = \{\gamma_i\}$ be $M \times 1$ vector of scaling parameters and $\alpha$ is a $k \times M$ matrix of regression-type coefficients. Vinod’s (P1998b) Theorem 1 proves that the sample estimate $(\hat{\alpha}\hat{\gamma})$ is a consistent estimate of $(\alpha\gamma)$. 

Figure 2: Comparison of the fits of demand function by OLS and by loess, showing advantages of loess in forecasting problems.
Vinod’s Theorem 2 proves that $\partial E(y|X)/\partial X$ is conveniently estimated by the matrix product $\alpha \gamma$, which is vector similar to the usual vector of regression coefficients.

### 11.5 Exercise (projection pursuit regression snippets)

1) What is the R function used to run a projection pursuit regression?

2) What is the R function used to create leads and lags of time series data?

3) What is the notation for $\gamma$ inside R function used to run a projection pursuit regression?

4) What is M? What is the criterion used to decide the choice of M? Why does it make sense?

5) Describe the final criterion used in choosing M in the snippet #R11.4.4.

**Answers:** 1) ppr 2) LeadsLags and embed 3) beta 4) M= Number of passes of the algorithm. The criterion is “sum of squares of all relevant residual autocorrelations.” In time series regressions, it is useful to have uncorrelated residuals after the fit. 5) After choosing the range of M values based on sum of squares of all relevant residual autocorrelations, the snippet computes the ratio of residual sum of squares (RSS) of comparable OLS with ppr. Note that the ratio is high if ppr shows great improvement over OLS.

\[
\text{Ratio} = \frac{\text{RSS(OLS)}}{\text{RSS(ppr)}}
\]

The idea then is to choose M to maximize this ‘Ratio’ within the range of M values based on minimizing sum of squares of all relevant residual autocorrelations.

### 11.6 Exercise (Icecream data, projection pursuit regression)

Estimate a demand function for icecream using the data called ‘Icecream’ of the ‘Ecdat’ package by ordinary least squares (OLS). Regress ‘cons’ on ‘income,’ and ‘temperature’ after using 1 lead and 1 lag and on ‘price.’ Use the methods in snippet #11.4.4 for projection pursuit regression.
library(Ecdat)
data(Icecream);
Icecream=ts(Icecream, start=c(1951,1), frequency=12)
plot(Icecream)

con=Icecream[, "cons"]
inc=Icecream[, "income"]
pr=Icecream[, "price"]
tem=Icecream[, "temp"]
coef.original=coef(lm(cons~income+price+temp, data=Icecream))
coef.extractedData=coef(lm(con~inc+pr+tem))
rbind(coef.original, coef.extractedData)
The above code makes sure that we have the right data extracted (without using the attach command, which would not work for time series). Now copy and paste snipped #11.4.3 from the CD of the book into the memory of R. Now we apply the LeadsLags function to compute 1 lag and 1 lead series for both income and temperature series as follows with preceding character of y and t for the income and temperature.

```r
LeadsLags=function (y, nleads = 0, nlags = 0,
name.prefix = "yt")
#author Prof. H. D. Vinod, Fordham University, April 11, 2007
#purpose create variables for leads and lags
# without missing data
#name prefix must be a character, Use attach command
#to get names
{
  nn = nleads + nlags + 1
  # starty=start(y)[1]; freqy=starty[2]
  outy = embed(y, nn)
  np = paste(name.prefix, "+", sep = "")
  nm = paste(name.prefix, ";", sep = "")
  nam = rep(NA, nn)
  if (nleads > 0)
    nam[1:nleads] = paste(np, (nleads:1), sep = "")
  nam[nleads + 1] = name.prefix
  if (nlags > 0)
    nam[nn:(nn- nlags + 1)] = paste(nm, (nlags:1), sep = "")
  colnames(outy) = nam
  outy = as.data.frame(outy)
  # outy = as.ts(outy,start=starty-nlags, frequency=freqy)
  return(outy) #comment out this line and use next line
  #list(oy=outy, nam=nam) # gets out name list
}
Linc=LeadsLags(inc,1,1,"y")
Linc=ts(Linc, start=c(1951,2), frequency=12)
```
Litem = LeadsLags(tem, 1, 1, "t")
Litem = ts(Litem, start = c(1951, 2), frequency = 12)
mys = start(ts.intersect(Icecream, Linc))
mye = end(ts.intersect(Icecream, Linc))
rbind(mys, mye)
  [, 1] [, 2]
# mys 1951 2
# mye 1953 5
cons = window(con, start = mys, end = mye)
Lincome = window(Linc, start = mys, end = mye)
Ltemp = window(Ltem, start = mys, end = mye)
price = window(pr, start = mys, end = mye)

rang = 5:20; j = 0
ac = rep(NA, length(rang))
for (i in rang){j = j + 1
  reg = ppr(cons ~ Lincome + Ltemp + price,
            nterms = i, max.terms = 25)
  mm = mean(abs(acf(reg$residuals, type = "correlation", plot = F)$acf))
  ac[j] = mm }
# end loop over i in the range=rang

plot(rang, ac, typ = "l", main = "Choosing the number to passes in projection pursuit",
     xlab = "Number of passes or M", ylab = "Sum of squares of all relevant residual autocorrelations")
cbind(rang, ac)

rang   ac
[1,]  5 0.1619139
[2,]  6 0.2279687
[3,]  7 0.1775868
[4,]  8 0.2289355
[5,]  9 0.2018908
[6,] 10 0.1737670
[7,] 11 0.2392101
[8,] 12 0.2104980
[9,] 13 0.2129238
Choosing the number to passes in projection pursuit

Figure 4: Choice of $M=$number of passes in Projection Pursuit, Icecream data, 1 lead 1 lag for income and temperature
The above figure and table suggests M=14 to 18 as the desirable range of number of passes.

dols=lm (cons~Lincome+Ltemp+price)
mdols=mean(abs(acf(dols$residuals,type="correlation", plot=F)$acf))
print(c("mean of sum of sq of acf autocorrelations if OLS is used=ssrac=", mdols),q=F)
dox=dols$coef
dol=dols$coef[2:length(dox)] #selected coefficients
nterms=c(14:18)
for (nter in nterms) {#loop over only 2 choices over # M=No. of passes
  print(c("M=",nter),q=F)
  reg2=ppr(cons~Lincome+Ltemp+price, nterms=nter,max.terms=25)
  mppr=mean(abs(acf(reg2$residuals,type="correlation", plot=F)$acf))
  print(c("mean of sum of sq of acf autocorrelations if PPR is used=ssrac=", mppr),q=F)
alp=as.matrix(reg2$alpha)
bet=as.numeric(reg2$beta)
ppcoef=alp %*%bet
print(cbind(ppcoef,dol))
#number of lags used in acf is defaulted at
# 10*log10(N/m) where N is the number of observations
#and m the number of series.
print("following measures how close proj pursuit is to OLS")
\begin{verbatim}
print(sum((ppcoef-dol)^2))
rsddols=sum(resid(dols)^2)
rsppr=sum(resid(reg2)^2)
print(c("resid sum of sq ols, ppr, their
ratio",rsddols,rssppr, (rsddols/rssppr)),q=F)
\end{verbatim}

In the notation here we use the L prefix when the LeadsLags function is used. Leads are denoted by +1 and lags by -1, income has the additional suffix "y" and temperature has the additional suffix "t".

[1] M= 14
[1] mean of sum of sq of acf autocorrelations if PPR is\n used=ssrac=
[2] 0.178599812437463
dol
Lincomey+1  0.010608511  0.0011264742
Lincomey   -0.002899494  -0.0014409930
Lincomey-1  0.013355575  0.0029051926
Ltempt+1    0.004302561  0.0009806989
Ltempt     -0.029046897  -0.0033636839
Ltempt-1   -0.010559998  -0.0013586713
price      -0.048220970  -0.1797312779
[1] "following measures how close proj pursuit is to OLS"
[1] 0.01825153
[1] resid sum of sq ols, ppr, their\nratio
[2] 0.0095275748484257
[3] 1.89329658345618e-07
[4] 50322.6749135799
[1] M= 15
[1] mean of sum of sq of acf autocorrelations if PPR is\n used=ssrac=
[2] 0.198063731478373
dol
Lincomey+1  0.010630058  0.0011264742
Lincomey   -0.002905080  -0.0014409930
Lincomey-1  0.013341084  0.0029051926
Ltempt+1    0.004297286  0.0009806989
Ltempt     -0.029062664  -0.0033636839
Ltempt-1   -0.010573362  -0.0013586713
price      -0.048096877  -0.1797312779
\end{verbatim}
"following measures how close proj pursuit is to OLS"

0.01828533

resid sum of sq ols, ppr, their ratio
0.0095275748484257
6.48785315554286e-09
1468525.04518939

M= 16

mean of sum of sq of acf autocorrelations if PPR is used=ssrac=
0.148497785324006
dol
Lincomey+1 0.010647404 0.0011264742
Lincomey -0.002884579 -0.0014409930
Lincomey-1 0.013306075 0.0029051926
Ltempt+1 0.004273071 0.0009806989
Ltempt 0.029104027 0.0033636839
Ltempt-1 -0.010596330 -0.0013586713
price -0.048075768 -0.1797312779

"following measures how close proj pursuit is to OLS"
0.01829282

resid sum of sq ols, ppr, their ratio
0.0095275748484257
2.35922448957214e-08
403843.504106453

M= 17

mean of sum of sq of acf autocorrelations if PPR is used=ssrac=
0.160634263755075
dol
Lincomey+1 0.010649005 0.0011264742
Lincomey -0.002882951 -0.0014409930
Lincomey-1 0.013302902 0.0029051926
Ltempt+1 0.004270620 0.0009806989
Ltempt 0.029108030 0.0033636839
Ltempt-1 -0.010598442 -0.0013586713
price -0.048075768 -0.1797312779

"following measures how close proj pursuit is to OLS"
0.01829282
2.66398289691815e-09
3576439.94616022
M= 18
mean of sum of sq of acf autocorrelations if PPR is
used=ssrac=
0.214833797641628
dol
Lincomey+1 0.010649883 0.0011264742
Lincomey -0.002881280 -0.0014409930
Lincomey-1 0.013300965 0.0029051926
Ltempt+1 0.004270337 0.0009806989
Ltempt 0.029108841 0.0033636839
Ltempt-1 -0.010598637 -0.0013586713
price -0.048211551 -0.1797312779
"following measures how close proj pursuit is to OLS"
0.01825729
resid sum of sq ols, ppr, their
ratio
0.0095275748484257
3.97841118562472e-09
2394819.038025
Clearly M=17 has the best fit compared to OLS suggested by the ratio of
residual sum of squares for OLS divided by residual sum of squares for pro-
jection pursuit regression. The larger the ratio, the greater the improvement
over OLS. When M=17 this ratio is 3576440 rounded to the nearest integer.
It is larger than M=18 value of 2394819 and M=16 value of 403843.5.

reg2=ppr(cons˜Lincome+Ltemp+price,
nterms=17,max.terms=17)
alp=as.matrix(reg2$alpha)
bet=as.numeric(reg2$beta)
ppcoef=alp %*%bet

Thus the final regression coefficients by projection pursuit method are:

[,1]
Lincomey+1 0.010646577
Lincomey -0.002880960
Lincomey-1 0.013303366
Ltemp+1  0.004275233
Ltemp    0.029100650
Ltemp-1  -0.010594035
price    -0.048192854

where Lincomey+1 is next month’s income, Lincomey is current monthly income, Lincomey-1 is past month’s income, Ltemp+1 is next month’s temperature, Lincomey is current month’s temperature and Lincomey-1 is past month’s temperature. The effect of price of icecream is negative consistent with the law of demand. Effect of current temperature on demand for icecream is positive, as expected.

11.7 Exercise (Icecream data, projection pursuit regression, confidence intervals using meboot)

Estimate a demand function for icecream using the data called ‘Icecream’ of the ‘Ecdat’ package by ordinary least squares (OLS). Regress ‘cons’ on ‘income,’ after using 1 lead and 1 lag, and on ‘temperature’ after using 0 lead and 1 lag, and also on ‘price.’ Use the methods in snippet #11.4.4 for projection pursuit regression and the ME bootstrap of Chapter 9.

ANSWER Use the code for previous exercise as a template and make necessary changes. In the following some outputs are suppressed.

```r
rm(list = ls())  # clean up before start
library(Ecdat)
data(Icecream);
Icecream = ts(Icecream, start = c(1951, 1), frequency = 12)
plot(Icecream)
con = Icecream[, "cons"]
inc = Icecream[, "income"]
pr = Icecream[, "price"]
tem = Icecream[, "temp"]
coef.original = coef(lm(cons ~ income + price + temp, data = Icecream))
coef.extractedData = coef(lm(con ~ inc + pr + tem))
rbind(coef.original, coef.extractedData)

LeadsLags = function (y, nleads = 0, nlags = 0,

```
name.prefix = "yt")
#author Prof. H. D. Vinod, Fordham University, April 11, 2007
#purpose create variables for leads and lags
#without missing data
#name prefix must be a character, Use attach command
#to get names
{ nn = nleads + nlags + 1
  name.prefix="0"
  #if(nlags==0) name.prefix="0"
  # starty=start(y)[1]; freqy=starty[2]
  outy = embed(y, nn)
  np = paste(name.prefix, "+", sep = "")
  nm = paste(name.prefix, "-", sep = "")
  nam = rep(NA, nn)
  if (nleads > 0)
    nam[1:nleads] = paste(np, (nleads:1), sep = "")
    nam[nleads + 1] = name.prefix
  if (nlags > 0)
    nam[nn:(nn- nlags + 1)] = paste(nm, (nlags:1), sep = "")
  colnames(outy) = nam
  outy = as.data.frame(outy)
  # outy = as.ts(outy,start=starty-nlags, frequency=freqy)
  return(outy) #comment out this line and use next line
  #list(oy=outy, nam=nam) # gets out name list
}

Inc=LeadsLags(inc,nleads=1,nlags=1,name.prefix="")
Inc=ts(Inc, start=c(1951,2), frequency=12)
Tem=LeadsLags(tem,nleads=0,nlags=1,name.prefix="")
Tem=ts(Tem, start=c(1951,2), frequency=12)
mys=start(ts.intersect(Icecream,Inc))
mye=end(ts.intersect(Icecream,Inc))
rbind(mys,mye)

cons=window(con,start=mys,end=mye)
Income=window(Inc,start=mys,end=mye)
Temp=window(Tem,start=mys,end=mye)
price=window(pr,start=mys,end=mye)

rang=5:20; j=0
ac=rep(NA,length(rang))
for (i in rang){j=j+1
reg=ppr(cons~Income+Temp+price,
nterms=i,max.terms=25)
mm=mean(abs(acf(reg$residuals,type="correlation",plot=F)$acf))
ac[j]=mm } #end loop over i in the range=rang

plot(rang, ac, typ="l", main="Choosing the number to passes
in projection pursuit",
xlab="Number of passes or M", ylab="Sum of squares of all
relevant residual autocorrelations")
cbind(rang,ac)

Note that residual autocorrelations are minimized at M=9 along the fifth row
in the following table. This suggests a neighborhood of desirable M values.
We also want good fit to data in this neighborhood.

<table>
<thead>
<tr>
<th>rang</th>
<th>ac</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5 0.2101341</td>
</tr>
<tr>
<td>2</td>
<td>6 0.1903678</td>
</tr>
<tr>
<td>3</td>
<td>7 0.2005913</td>
</tr>
<tr>
<td>4</td>
<td>8 0.1997831</td>
</tr>
<tr>
<td>5</td>
<td>9 0.1593866</td>
</tr>
<tr>
<td>6</td>
<td>10 0.1917622</td>
</tr>
<tr>
<td>7</td>
<td>11 0.2057589</td>
</tr>
<tr>
<td>8</td>
<td>12 0.1983636</td>
</tr>
<tr>
<td>9</td>
<td>13 0.2124799</td>
</tr>
<tr>
<td>10</td>
<td>14 0.2091264</td>
</tr>
<tr>
<td>11</td>
<td>15 0.1999216</td>
</tr>
<tr>
<td>12</td>
<td>16 0.2066412</td>
</tr>
<tr>
<td>13</td>
<td>17 0.2568586</td>
</tr>
<tr>
<td>14</td>
<td>18 0.1994455</td>
</tr>
<tr>
<td>15</td>
<td>19 0.1793957</td>
</tr>
<tr>
<td>16</td>
<td>20 0.1937225</td>
</tr>
</tbody>
</table>

The figure suggests M=7:11 as the range.
Choosing the number to passes in projection pursuit

Figure 5: Choice of $M =$ number of passes in Projection Pursuit, Icecream data, 1 lead 1 lag for income and 1 lag for temperature
dols=lm (cons~Income+Temp+price)
mdols=mean(abs(acf(dols$residuals,type="correlation", plot=F)$acf))
print(c("mean of sum of sq of acf autocorrelations if OLS is used=ssrac=",
mdols),q=F)
dox=dols$coef
dol=dols$coef[2:length(dox)] #selected coefficients

nterms=c(7:11)
for (nter in nterms) {#loop over only 2 choices over
# M=No. of passes
print(c("M=",nter),q=F)
reg2=ppr(cons~Income+Temp+price,
nterms=nter,max.terms=25)
mppr=mean(abs(acf(reg2$residuals,type="correlation", plot=F)$acf))
print(c("mean of sum of sq of acf autocorrelations if PPR is used=ssrac=",
mppr),q=F)
alp=as.matrix(reg2$alpha)
bet=as.numeric(reg2$beta)
ppcoef=alp %*%bet
print(cbind(ppcoef,dol))
#number of lags used in acf is defaulted at
# 10*log10(N/m) where N is the number of observations
#and m the number of series.
print("following measures how close proj pursuit is to OLS")
print(sum((ppcoef-dol)^2))
rssdols=sum(resid(dols)^2)
rssppr=sum(resid(reg2)^2)
print(c("resid sum of sq ols, ppr, their ratio",rssdols,rssppr, (rssdols/rssppr)),q=F)
}

Looking at the output above (Suppressed for brevity) we choose M=11 as giving the best fit in the desirable range determined from residual autocorrelations above.
reg2=ppr(cons~Income+Temp+price,
nterms=11, max.terms=11)
alp=as.matrix(reg2$alpha)
bet=as.numeric(reg2$beta)
ppcoef=alp %*%bet

ppcoef

[,1]
Income0+1  0.008525269
Income0   -0.003868211
Income0-1  0.013197429
Temp0     0.029414954
Temp0-1   -0.011503937
price     -0.065995008

This shows that the coefficient of price has the right sign (negative) and that of current month temperature also has the right sign (positive).

Now we turn to use the ME bootstrap discussed in Chapter 9. We begin by writing an R function called `bstar.icecream` to estimate the resampled slope coefficients. In that function we create 999 ensembles (resamples) for each variable involved separately and then compute leads and lags. Note that we need to be careful about the starting values ‘mys’ and ending values ‘mye’ for the months included in the regression. This is done conveniently in R by the function ‘ts.intersect.’ We then use the ‘window’ function in R to select the correct subset of data. Only then we are ready to use the ‘ppr’ function to do the projection pursuit regression. Finally we use the matrix multiplication ‘alp %*

library(meboot)
# new R function
#
# bstar.icecream <- function(con, inc, tem, pr,
# level = 0.95, bigJ = 999, seed1 = 135) {
# set.seed(seed1)
# semy <- meboot(x = con, reps = bigJ)$ensemble
# semx1 <- meboot(x = inc, reps = bigJ)$ensemble
# semx2 <- meboot(x = tem, reps = bigJ)$ensemble
# semx3 <- meboot(x = pr, reps = bigJ)$ensemble
# n <- NROW(con)
# m <- length(theta.cobbd(y, x1, x2))
# if(m!=1) stop("too many parameters in theta")
bb <- matrix(NA, 6, bigI)
for(j in 1:bigJ) {
    yy <- semy[,j]
    xx1 <- semx1[,j]
    xx2 <- semx2[,j]
    xx3 <- semx3[,j]
    Inc=LeadsLags(xx1,nleads=1,nlags=1,name.prefix="")
    Inc=ts(Inc, start=c(1951,2), frequency=12)
    Tem=LeadsLags(xx2,nleads=0,nlags=1,name.prefix="")
    Tem=ts(Tem, start=c(1951,2), frequency=12)
    mys=start(ts.intersect(Icecream,Inc))
    mye=end(ts.intersect(Icecream,Inc))
    #rbind(mys,mye)
    y=window(yy,start=mys,end=mye)
    Income=window(Inc,start=mys,end=mye)
    Temp=window(Tem,start=mys,end=mye)
    price=window(xx3,start=mys,end=mye)
    reg2=ppr(cons~Income+Temp+price,
              nterms=11, max.terms=11)
    alp=as.matrix(reg2$alpha)
    bet=as.numeric(reg2$beta)
    ppc=alp *%*%bet
    bb[,j] <- ppc
}
return(bb)
}
# end R function
bb=bstar.icecream(con, inc, tem, pr)

We have implemented maximum entropy bootstrap from the package ‘me-boot’ [1]. We constructed a large matrix ‘bb’ containing the 999 sets of 6 coefficients for 6 slopes. Our next task is to analyze these resampled slopes to determine statistical significance. A graphical view is provided by the package ‘hdrcde’ [2] which is called as follows.

library("hdrcde")
Figure 6: Approximation to the sampling distribution of the slope of the price regressor using ‘Icecream’ dataset

```r
hdr.den(bb[,6], main = expression(Highest ~ density ~ region ~ b[6] ~ price ~ of ~ Ice-cream))
```

A similar figure with two regions associated with a bi-modal resampling density is seen for all slope coefficients, although the regions vary.

```r
nam=dimnames(ppcoef)[[1]]
for (i in 1:6){
  print(c("coefficient No.", i, nam[i]), quote=FALSE)
  #Percentile.bi=rep(NA,2)
  #Refined.bi=rep(NA,2)
  #hdr.bi=rep(NA,4)
  Percentile.bi <- quantile(bb[i,, c(0.025, 0.975), type = 8)
```

21
```r
Refined.bi <- null.ci(bb[i,])
hdr.bi = hdr(bb[i,])$hdr[2,]
# print(hdr.bi)[1:4] #$
print(rbind(Percentile.bi, Refined.bi, hdr.bi[1:2], hdr.bi[3:4]))
}

[1] coefficient No. 1    Income0+1
         2.5%     97.5%
Percentile.bi -0.003353031  0.002690394
Refined.bi -0.003780426  0.002515542
set1.hdr -0.004334976  -0.004149467
set2.hdr -0.003061027  -0.003024826
[1] coefficient No. 2    Income0
         2.5%     97.5%
Percentile.bi -0.004404508  0.003358852
Refined.bi -0.004348732  0.003481973
set1.hdr -0.003689621  -0.003205159
set2.hdr -0.002964835  0.003064647
[1] coefficient No. 3    Income0-1
         2.5%     97.5%
Percentile.bi -0.000844007  0.011705063
Refined.bi -0.003818621  0.006998654
set1.hdr -0.001560639  -0.001383442
set2.hdr -0.001132406  0.005052956
[1] coefficient No. 4    Temp0
         2.5%     97.5%
Percentile.bi  1.420858e-05  0.015250117
Refined.bi -1.791964e-03  0.009861290
set1.hdr -1.072812e-03  -0.000667466
set2.hdr -4.266164e-04  0.007266549
[1] coefficient No. 5    Temp0-1
         2.5%     97.5%
Percentile.bi -0.005662539  0.003408552
Refined.bi -0.003515203  0.002142920
set1.hdr -0.005705231  -0.005652005
set2.hdr -0.004448467  -0.004420423
[1] coefficient No. 6    price
         2.5%     97.5%
```

22
Since the highest density region has two modes, (See attached figure) there are two components in the 95% confidence intervals. The actual interval is a union of two sets. We report the two sets of intervals along two lower lines. Note that since both percentile and refined intervals for all 6 coefficients do contain the zero, all coefficients are statistically insignificant (NOT significantly different from zero. The ‘hdr’ interval for the coefficient for future income (lead=1) denoted by ‘Income0+1’ suggests significantly negative effect. Similarly the ‘hdr’ interval for lag=1 variable for temperature denoted by ‘Temp0-1’ suggests significantly negative effect in the sense that zero is outside these intervals. Perhaps, the upper limits -0.003 and -0.004 do not seem convincingly negative in light of other evidence.

References


